Report Name : Analyze the images if they contain real data or not

Course Name :Encryption Theory (Comp 438)

Teacher: Dr. Mohammed Al-Khanafsah

Students:

1- Tareq khanfar – 1200265

Contents

Introduction

The main function of the project is to help analyze and identify images that contain real data. The project accomplishes this by using the LSB algorithm to embed and extract data within images, and then checking whether the image contains data that is meaningful and relevant using the Jaccard, Minimum Edit Distance, and N-gram algorithms.

Technical Description :

Assume that Alice he want to send a secret message inside the image to BOB , the output will be a stegno. Image ,

Suppose there is a man-in-the-middle attack and he takes the Image

My project upload the image, then passes it to the LSB algorithm to extract data from it, and then at the beginning it is checked whether the information is Dummy or not through the Jaccard algorithm. After that, based on the percentage that will be obtained, a decision will be made if this information is dummy or not. If it is not, it will be passed to the *Minimum edit distance algorithm*, and then it will be compared with the values in the corpus using (N- gram Algorithm) . And this is repeated on all the values in the corpus until we get the highest percentage, and then based on the percentage, the appropriate decision will be taken if this information is really meaningful information or not

In addition to Alice and Bob, the system may also involve other users such as the administrator who manages the system or potential attackers who may attempt to intercept the transmission. It's important to consider the security of the system from the perspective of all the users involved, and to implement appropriate safeguards to protect the data being transmitted.

Recent research and artical that work in same field:

"A Deep Learning Approach for Steganalysis of Color Images" by Kaoutar El Maghraoui and Abdellatif Ennaji is a research article that proposes a deep learning approach for detecting hidden information in color images using a convolutional neural network (CNN).

The authors of the article argue that traditional steganalysis methods may not be effective against modern steganography techniques, which can be designed to be very difficult to detect. To overcome this limitation, they propose a deep learning approach using a CNN.

The CNN is trained on a large dataset of color images that have been either modified to contain hidden information or left unmodified. The CNN learns to identify patterns in the images that are indicative of hidden information. Once the CNN is trained, it can be used to classify new images as either containing hidden information or not.

The authors of the article report that their deep learning approach outperformed traditional steganalysis methods on a number of benchmark datasets, and they argue that the approach shows promise for detecting hidden information in real-world applications.

METHODOLOGY :

1-LSB ALGORITHM

Introduction:

The Least Significant Bit (LSB) algorithm is a steganographic method used to hide information within an image or audio file by modifying the least significant bits of the pixel values or audio samples. The human eye and ear are not sensitive to these small changes in pixel values or audio samples, making it possible to hide secret information in plain sight.

Overview of the Algorithm:

The LSB algorithm works by replacing the least significant bit of each pixel value or audio sample with a bit from the secret message. The least significant bit is the binary digit in the ones place, which represents the smallest possible value. For example, the number 5 in binary is 101, where the 1 in the ones place is the least significant bit.

To hide a secret message using the LSB algorithm, the following steps can be followed:

1- Convert the secret message into binary format.
2- Open the image or audio file in which you want to hide the secret message.
3- For each pixel value or audio sample, replace the least significant bit with a bit from the secret message.
4- Save the modified image or audio file.

To extract the hidden message, the LSB algorithm is reversed by extracting the least significant bit from each pixel value or audio sample, and combining them to form the secret message in binary format.

Applications of the Algorithm:

The LSB algorithm is widely used in digital steganography to hide secret messages within images or audio files. It can be used for a variety of purposes, including:

1- Secret communication: The LSB algorithm can be used to hide secret messages within images or audio files, which can be shared over public channels without arousing suspicion.

2- Digital watermarking: The LSB algorithm can be used to embed a digital watermark within an image or audio file, which can be used to verify the authenticity of the file and prevent unauthorized copying.

3- Data hiding: The LSB algorithm can be used to hide sensitive data within images or audio files, which can be useful for storing data securely.

Limitations of the Algorithm:

While the LSB algorithm is a simple and effective way to hide secret messages, it has several limitations, including:

1- Low capacity: The amount of information that can be hidden using the LSB algorithm is limited by the number of pixels or audio samples in the file. Larger files can accommodate more information, but the hidden message may be more noticeable.

2- Susceptibility to compression: Images or audio files that are compressed using lossy compression algorithms can cause the hidden message to be lost.

3- Vulnerability to attacks: The LSB algorithm is vulnerable to various attacks, such as statistical analysis, which can be used to detect the presence of hidden messages.

Conclusion:

The LSB algorithm is a simple and effective way to hide secret messages within images or audio files. It has several applications in digital steganography, including secret communication, digital watermarking, and data hiding. However, it also has limitations, including low capacity, susceptibility to compression, and vulnerability to attacks. Despite these limitations, the LSB algorithm remains a popular method for hiding secret messages in plain sight.

2- N-gram Algorithm:

N-grams are consecutive sequences of words in a text or speech, where "n" denotes the number of words in each sequence. These are widely used in natural language processing (NLP) to model the probability of the next word in a sequence based on the preceding words. The most common type of n-gram is a bigram (2-gram), which represents pairs of consecutive words. However, n-grams can also represent longer sequences, such as trigrams (3-grams) or 4-grams.

To generate n-grams, the text is first tokenized by breaking it into individual words or tokens. This involves using techniques such as splitting text on whitespace or punctuation, or using more advanced natural language processing tools like stemming and lemmatization. Once the text is tokenized, n-grams are generated by identifying contiguous sequences of "n" words. For example, a bigram model would count the frequency of each pair of consecutive words in the text.

N-grams can be used for various NLP applications such as language modeling, information retrieval, and machine translation. However, there are some limitations, such as the sparsity problem, where many possible n-grams may never occur in the training data, and the lack of context beyond a fixed window of n words.

To efficiently count the frequency of n-grams, data structures such as hash tables or tries can be used. Once the n-gram frequency counts are calculated, they can be used to compute the probability of each n-gram, forming the basis of the n-gram language model.

Overall, n-gram analysis is a powerful tool for understanding the patterns and structures of natural language and can be applied to a wide range of NLP and computational linguistics applications.

```
Function generate_ngrams(text, n):
  # tokenize the text into individual words
  words = tokenize(text)

  # initialize an empty list to hold the n-grams
  ngrams = []

  # loop over the words and generate n-grams
  for i from 0 to len(words) - n:
    ngram = []
    for j from i to i + n - 1:
      ngram.append(words[j])
    ngrams.append(ngram)

  # return the list of n-grams
  return ngrams
```

3-JACCARD ALGORITHM :

Note : this algorithm used in Model to check if the data is dummy or not, the reason of this because the time complexity for this algorithm is a O(1) , and the dummy data often be huge data , and is wrong pass it to MinEditAlgo. Because the time complexity O(n^2) . so will take the long time to obtain the result .

The Jaccard similarity algorithm is a technique used to compare the similarity between two sets of data. It calculates the similarity by dividing the size of the intersection of two sets by the size of the union of the same two sets. The result is a value between 0 and 1, where 0 indicates no similarity and 1 indicates a perfect match.

To illustrate the algorithm, let's say we have two sets of data:

Set A = {1, 2, 3, 4}

Set B = {3, 4, 5, 6}

To calculate the Jaccard similarity between Set A and Set B, we first find the intersection and union of the sets:

Intersection of Set A and Set B = {3, 4}

Union of Set A and Set B = {1, 2, 3, 4, 5, 6}

We then divide the size of the intersection by the size of the union:

Jaccard similarity = (Intersection of Set A and Set B) / (Union of Set A and Set B)

Jaccard similarity = 2 / 6 = 0.33

Therefore, the Jaccard similarity between Set A and Set B is 0.33.

The Jaccard similarity algorithm is commonly used in data mining and information retrieval tasks, such as document clustering, recommendation systems, and anomaly detection. It is a simple and efficient technique that can be applied to a wide range of data types, including text, images, and numerical data.

4- MINIMUM EDIT DISTANCE ALGORITHM

The Minimum Edit Distance (MED) algorithm is a technique used to measure the similarity between two strings. It calculates the minimum number of operations required to transform one string into another, where the allowed operations are insertion, deletion, and substitution of a single character. The result is a non-negative integer value that indicates the similarity between the two strings, where a smaller value indicates a higher similarity.

Example : Str1= "benan" and str2 = "bana"

ALGORITHM :

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j) + 1 \\ \operatorname{lev}_{a,b}(i,j-1) + 1 \\ \operatorname{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

XXXXXX	NULL	B	E	N	A	N
NULL	0	1	2	3	4	5
B	1	0	1	2	3	4
A	2	1	1	2	2	3
N	3	2	2	1	2	2
A	4	3	3	2	1	2

Result : Cost = 2

Therefore, the minimum edit distance between "benan" and "bana" is 2.

Pesudocode For Whole System

ALGORITHM :

STEGNO_ANALYSIS(IMAGE)

DATA_FROM_IMAGE ← EXTRACT_DATA(IMAGE)

PERCENTAGE ← JaccardAlgorithm(DATA_FROM_IMAG)

IF (PERCENTAGE > 30%) THEN :

Print ("NOT CONTAIN A DATA . I.E THE DATA IS DUMMY)

Else :

CURPOS ← Build_Curpos() // using N-gram Algorithm

MAX ← 0 , WORD ← NULL

FOR (VALUE IN CURPOS) THEN

Per. & word_Curpos ←MIN_EDIT_DISTANSE(VALUE , DATA_FROM_IMAGE)

IF (Per > MAX) THEN

MAX ←PERCENTAGE

WORD ← word_Curpos

END_IF

END_FOR

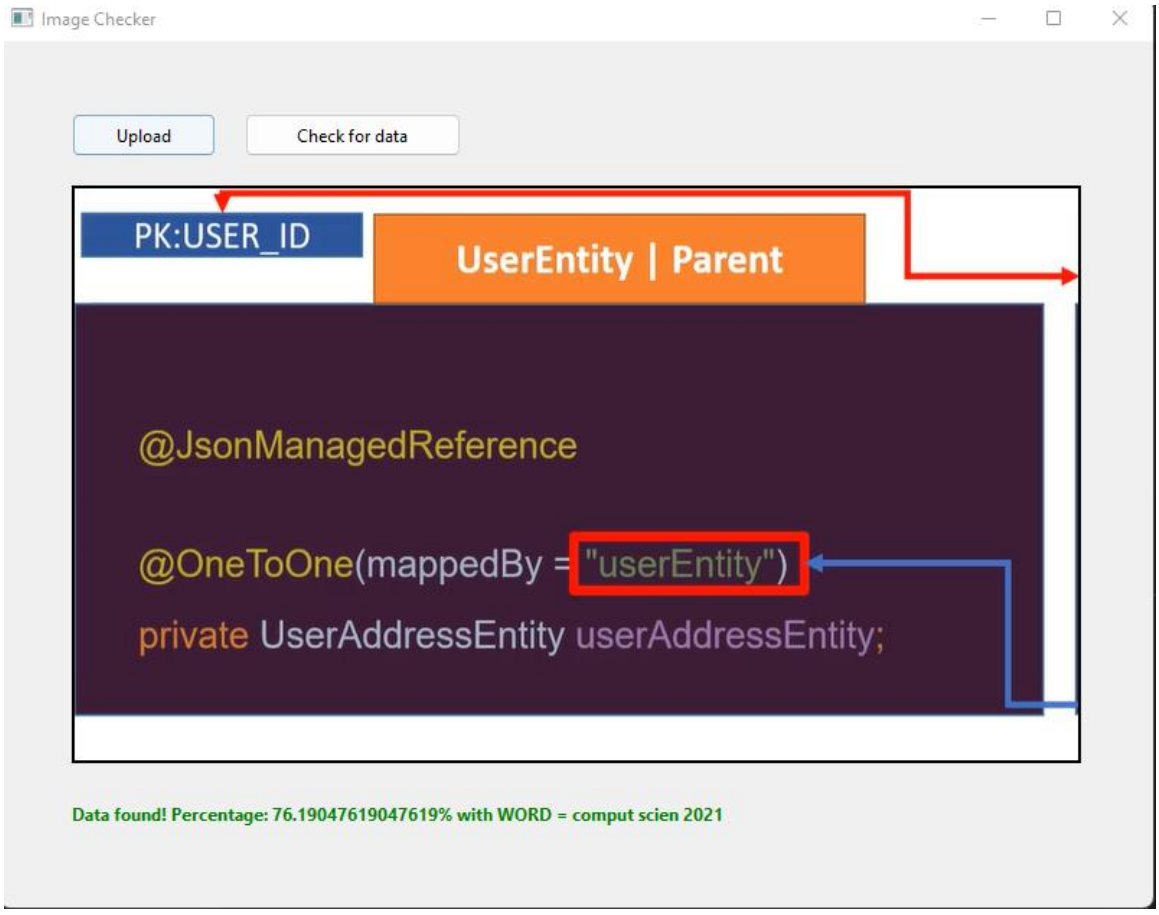
```
        IF (MAX > 50% ) THEN
                Print ("IMAGE IS CONTAIN DATA PER. %" +  . MAX )
                Print (" WITH WORD " + WORD )
        END _IF
END_FUNCTION
```

Result and discussion , results of applying idea and how the result difference from related solutions.

1-Assume that the image is contain these data “computer science 2022”

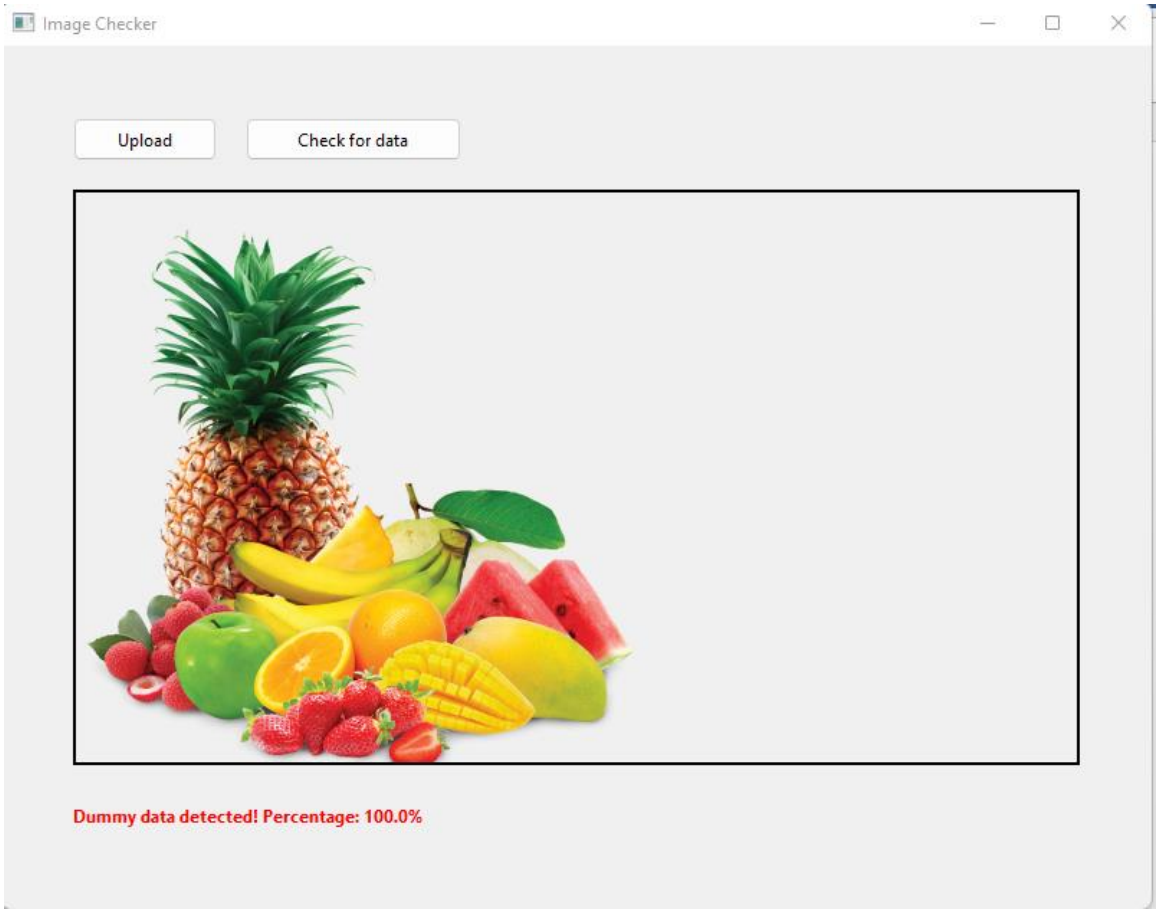
And the model not contain it exactly . but contain a “comput scien 2021”

The Result : DataFound ! The percentage 76.19 % with word “comput scien 2021”



2- assume that the Image not contain data , the result will be

“Dummy Data Detected! Percentage is 100%”



CONCLUSION

In conclusion, the project aimed to develop a system to analyze and identify images containing real data using the LSB algorithm to embed and extract data within images, and then verifying the meaningfulness of the information using the Jaccard, Minimum Edit Distance, and N-gram algorithms. The methodology section discussed the LSB algorithm in detail, including its applications and limitations in digital steganography. Recent research in the field of steganalysis was also presented.

The proposed system has the potential to improve the security of communication by allowing users to send hidden messages within images. However, it's important to consider the security of the system from the perspective of all users involved and implement appropriate safeguards to protect the data being transmitted.

Overall, the project contributes to the field of steganography and could be further developed to enhance its capabilities and address its limitations.

Future Work :

Yes, this can be done by developing the algorithms used into deep learning algorithms and providing the system with a very large dataset. From the interface of the stegno analyzer, this system can also be connected to the network, and thus the system automatically captures the images passing through the network and enters them into the system and then scans them. It sorts images that contain real data or not.